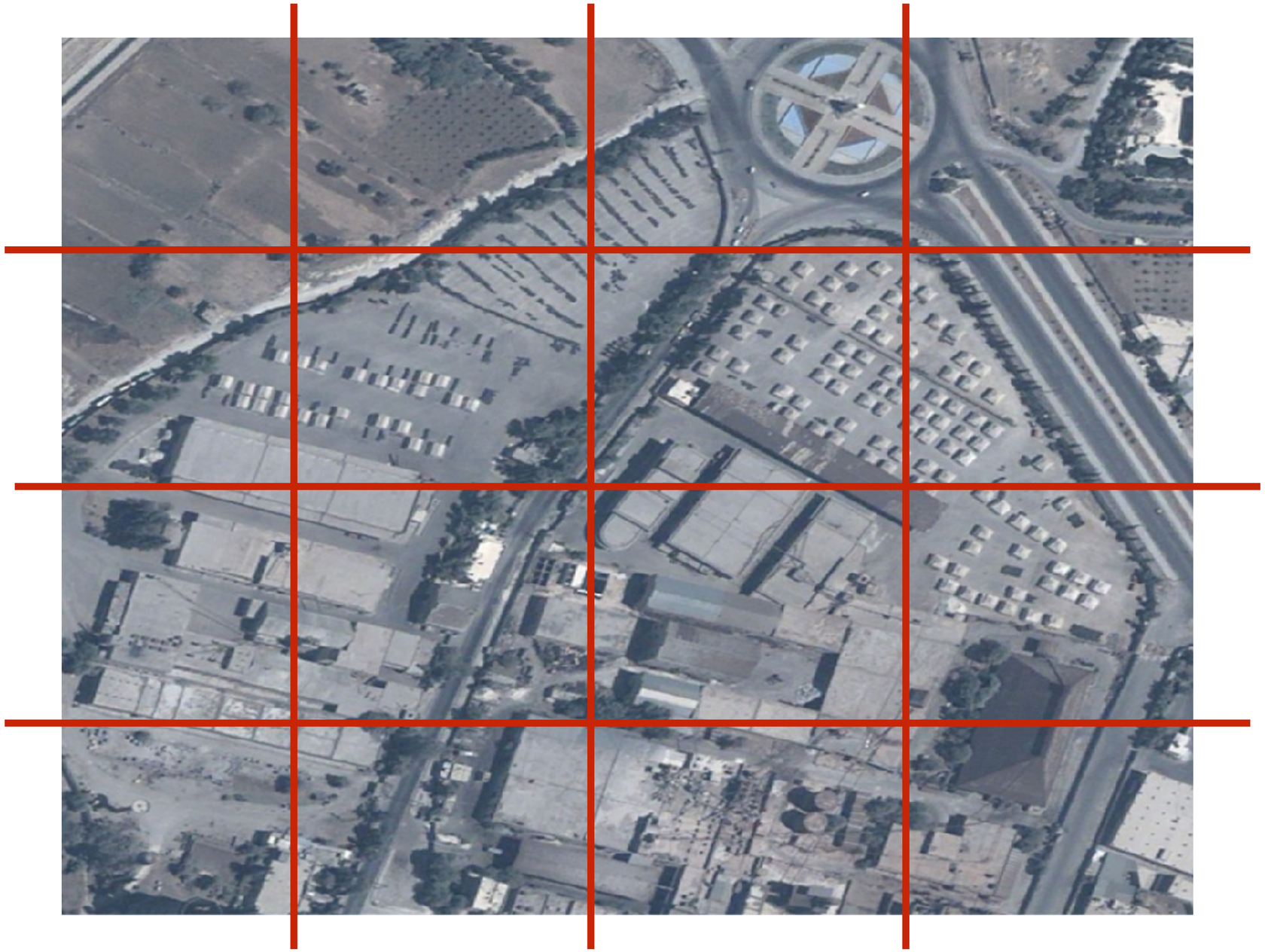
Disaster monitoring

Human computation model

- Changkun Ou
- Yifei Zhan
- Zhe Li

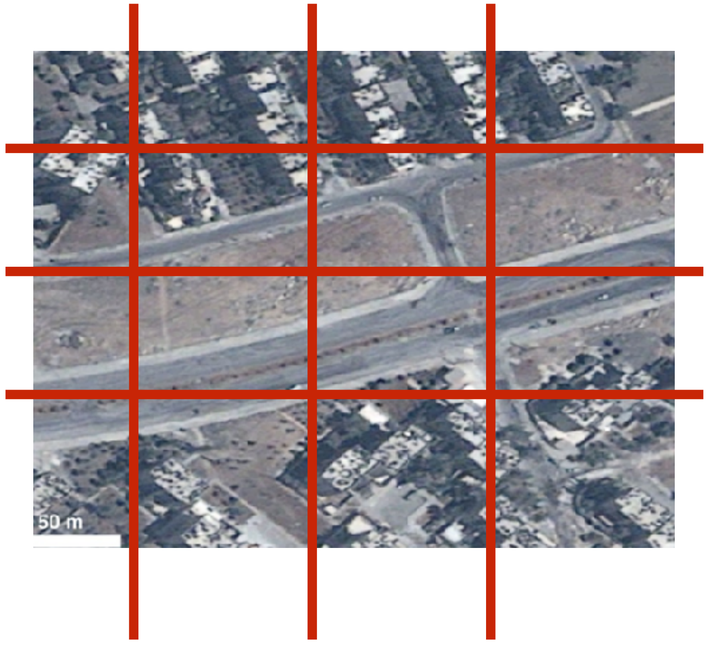
Unsolved issues since last time…

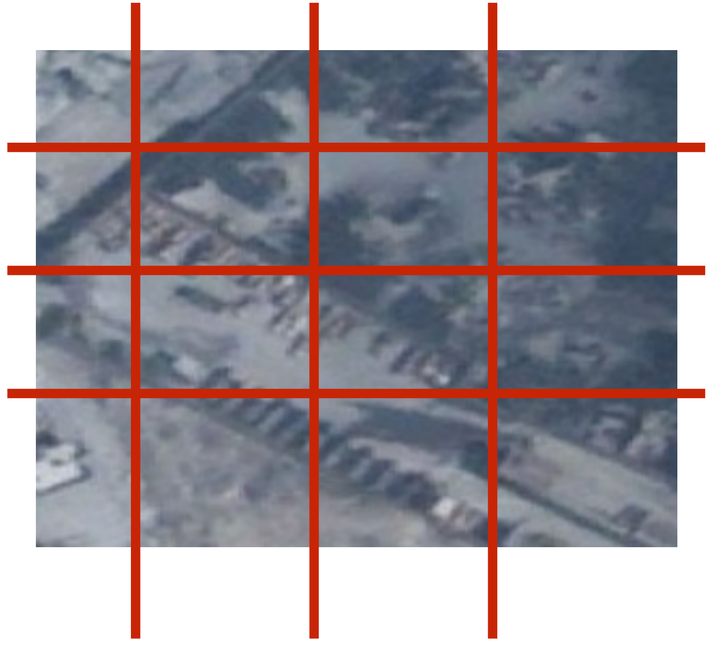
Too much information

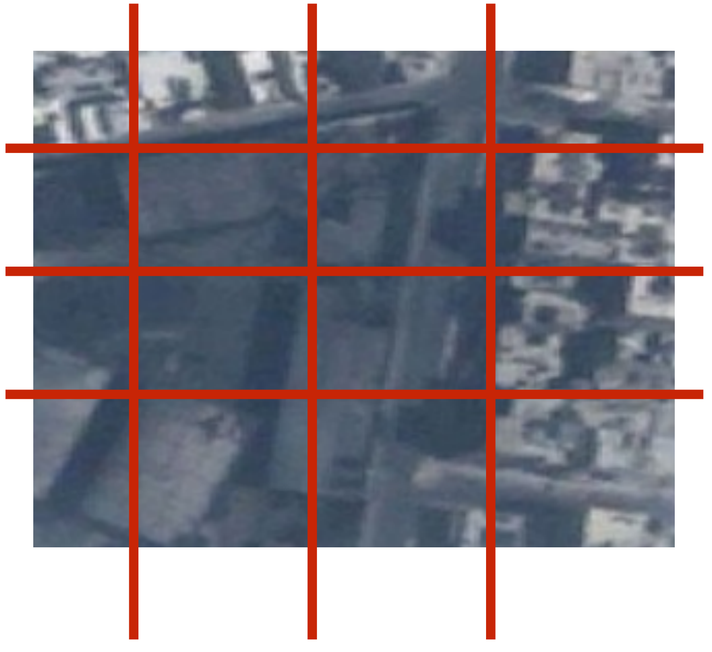


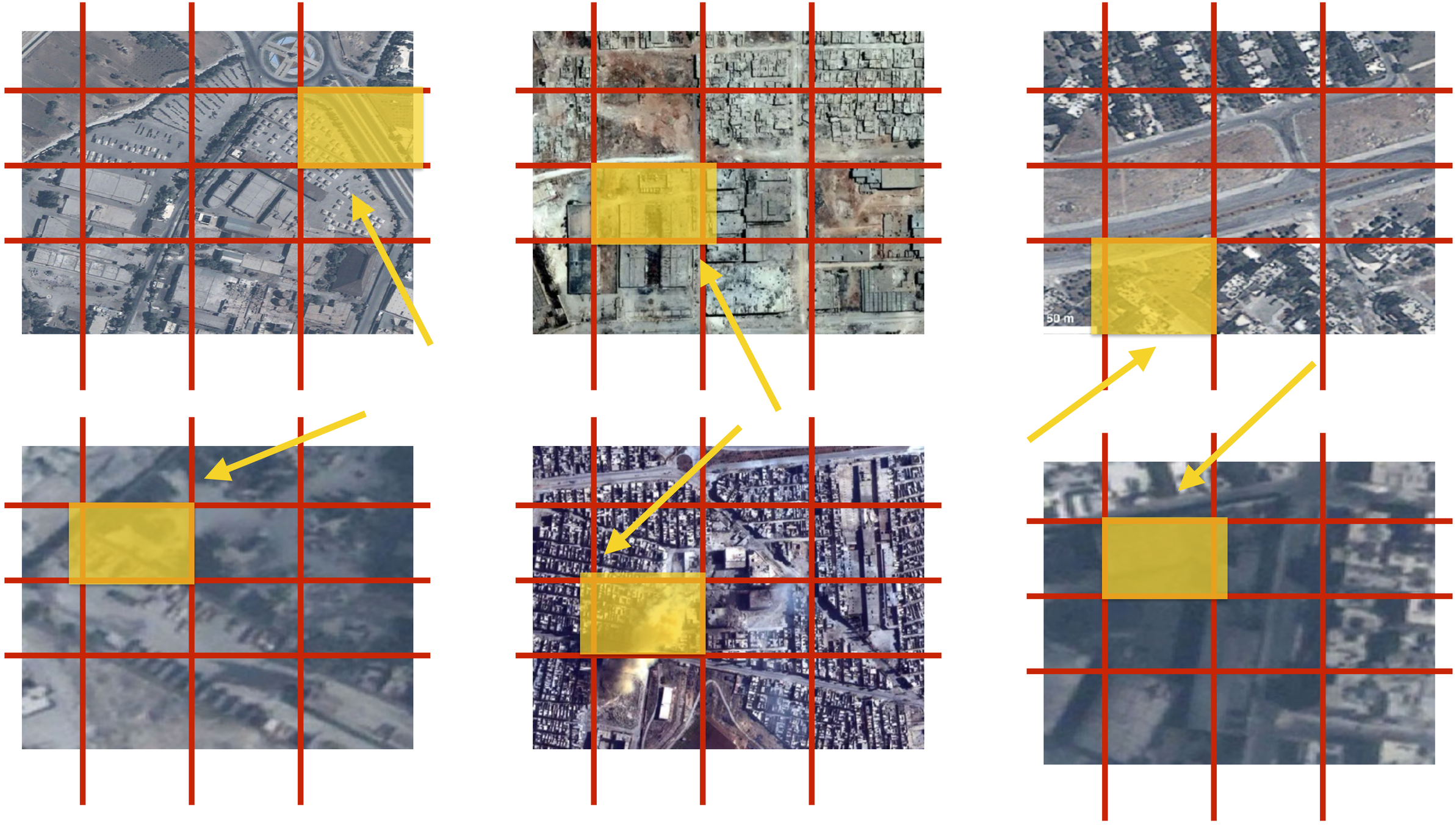
Cut it into fragment

ID to identify


50 m






50 m

Game with a purpose

Picture tagging

Pic 1.

Pic 2.

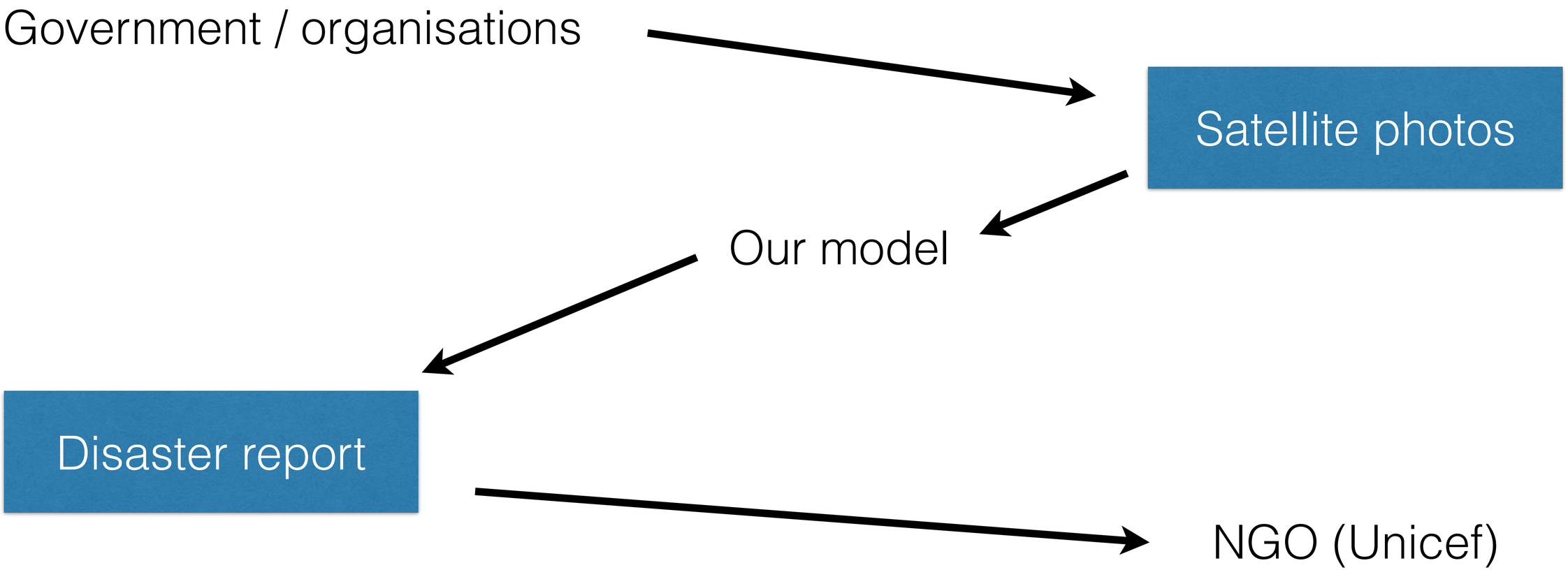
Pic 3.

…

Prototype

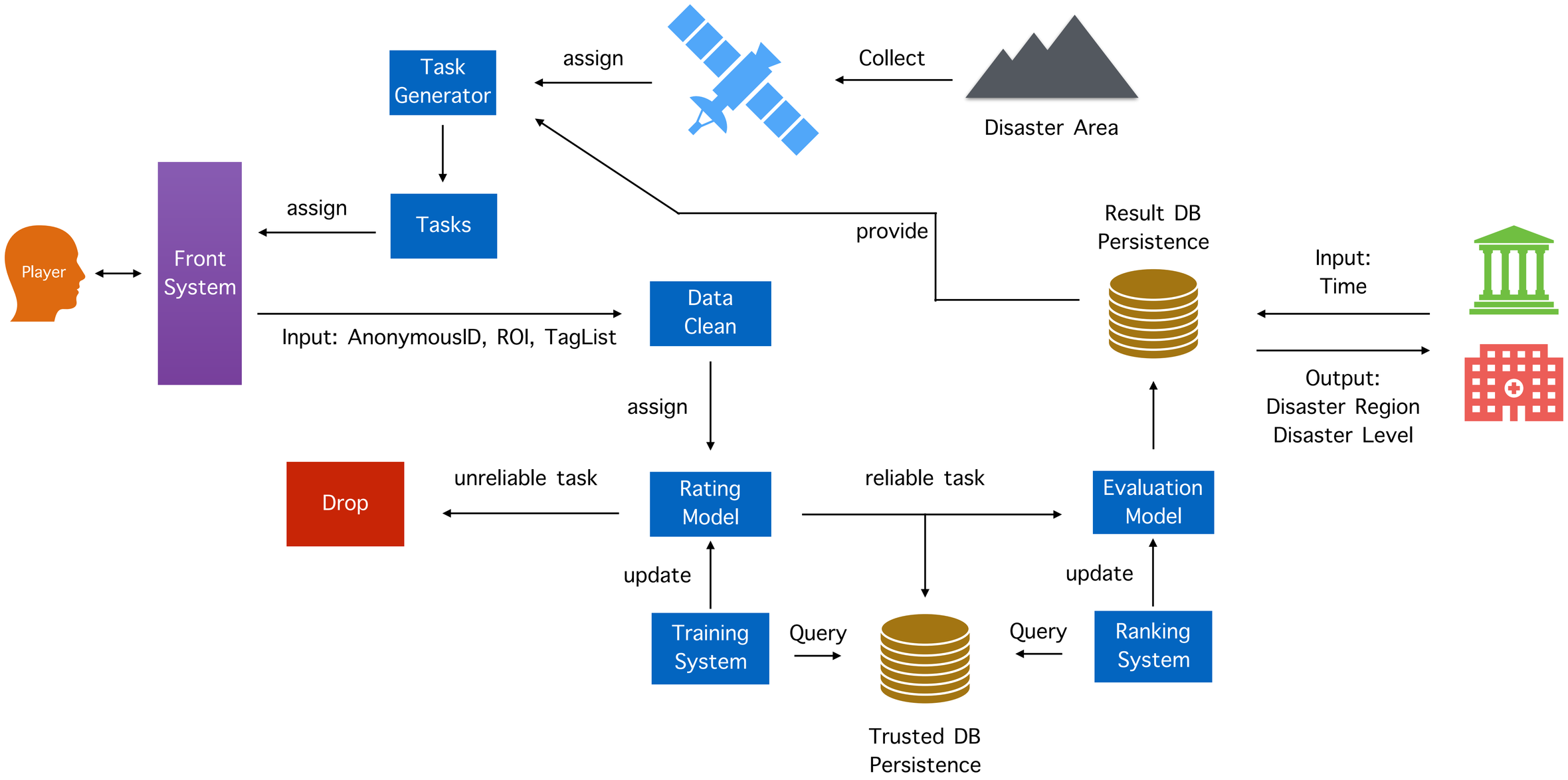
https://invis.io/2PC7IZRZM#/239461743_home-Page

Stakeholders

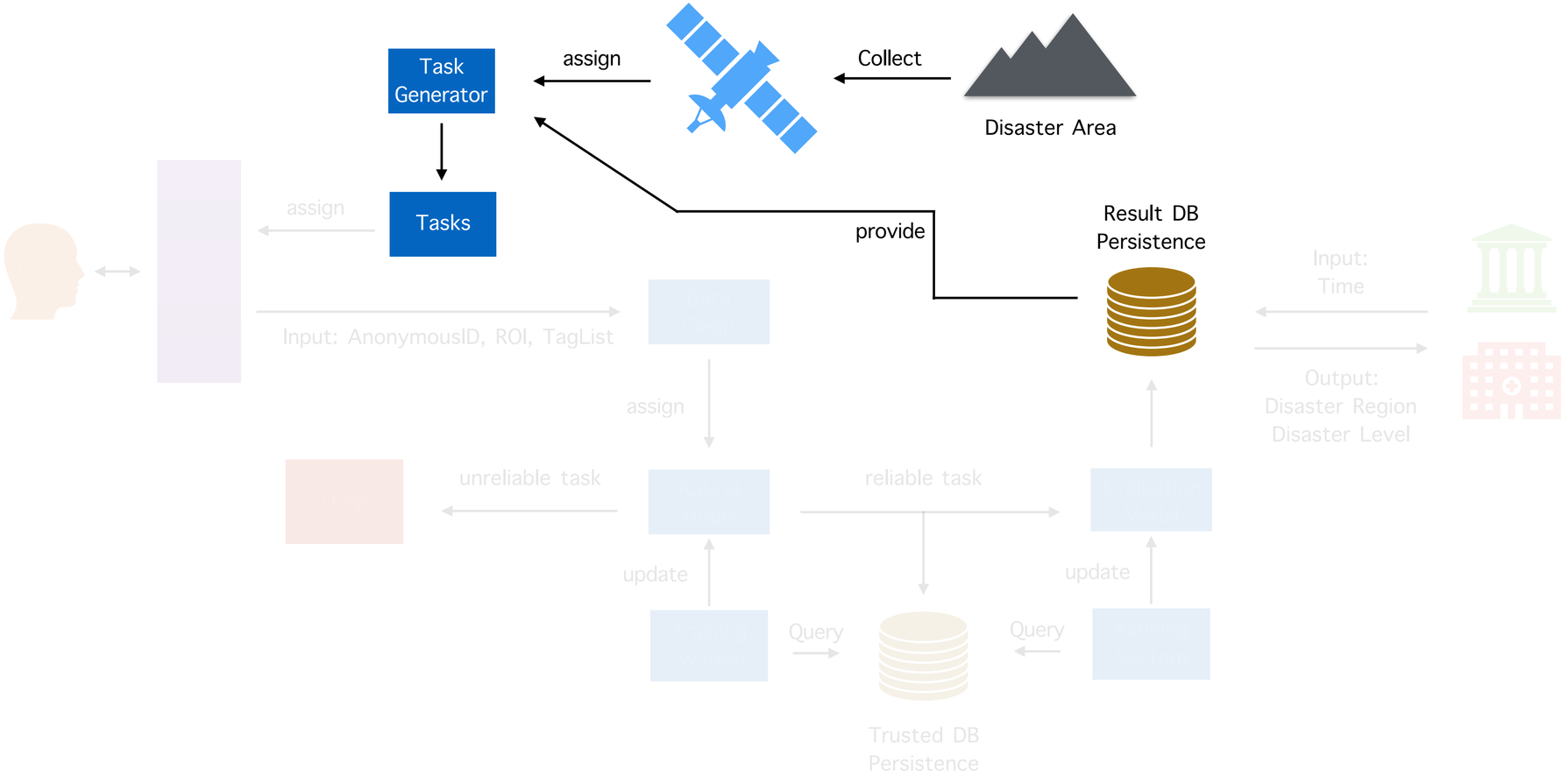

Government / organisations
Satellite photos
Our model
Disaster report
NGO (Unicef)

Input Aggregation Design & Models

Design Overview



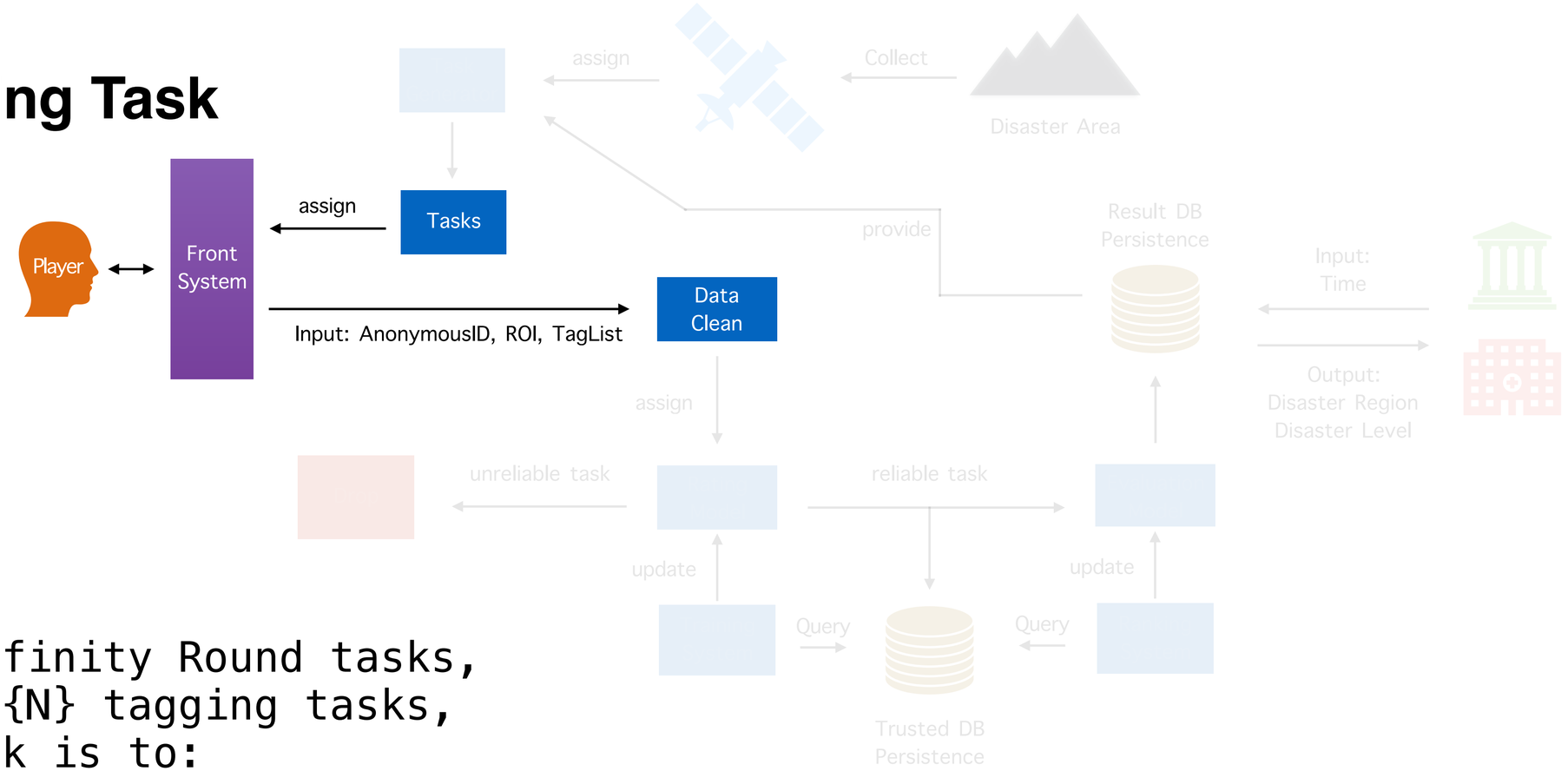
Task Generator



A task generator combines images from satellite and Result DB:

- Split a certain monitoring area image to pieces of images;
- Mix images from Result DB and pack as a Tagging Task which to be assigned to player;

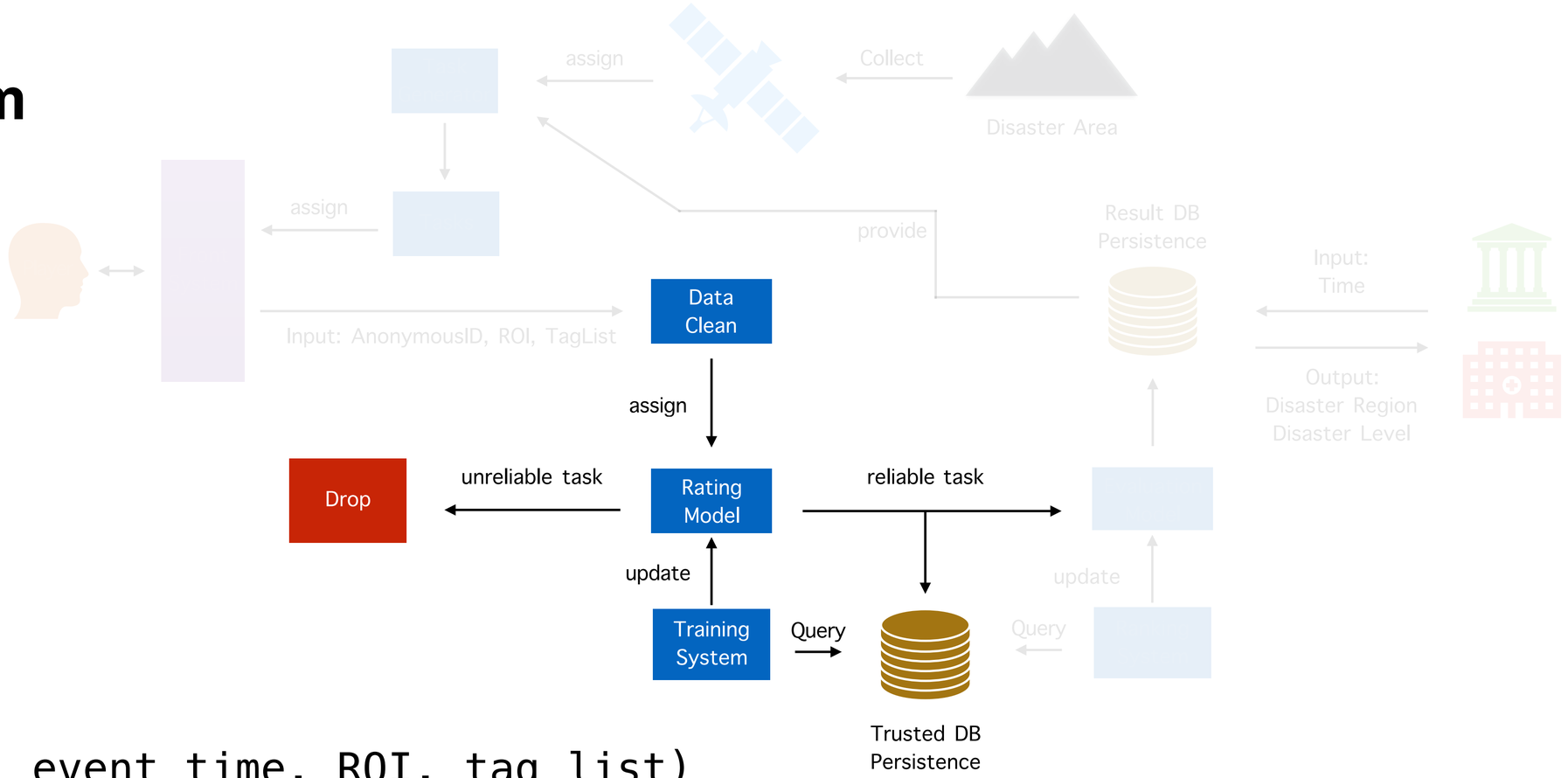
Game Rules & Tagging Task



A player can finish infinity Round tasks,
a Round task contains {N} tagging tasks,
the player tagging task is to:
- ***Select a Region Of Interests***(ROI) upon the presented satellite image;
- ***Tag the ROI*** from a provided tag list or input their own tag, the provided tag list contains: T_1, T_2, …, T_n, other(input needed)

Note that:
- A ROI is a sub-rectangle-window of a image;
- Multiple selections;
- Anyone can directly participant without registration, but system records an ID

Player Rating System



Player's input vector:
- (anonymous_id, image, event_time, ROI, tag_list)

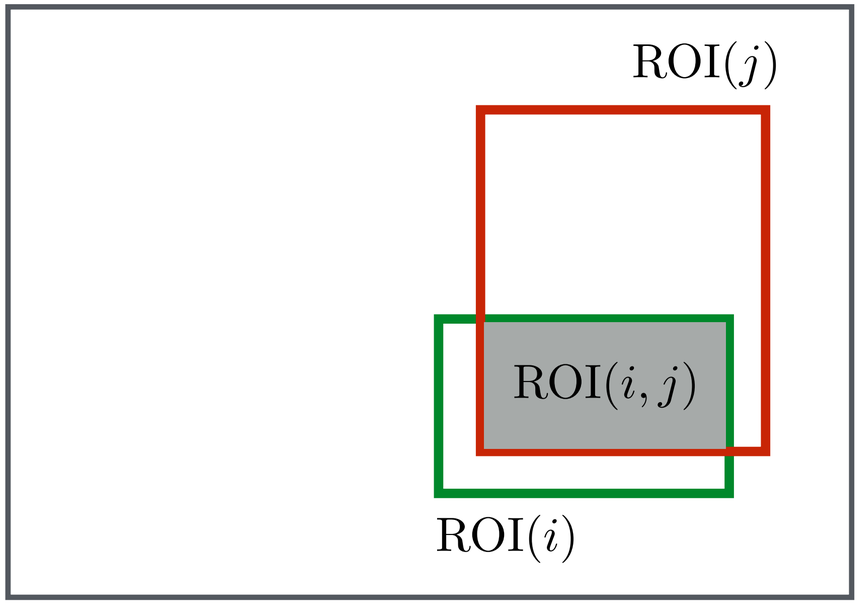
Model output:
- (anonymous_id, trust_value)

Note that:
- (anonymous_id, image, event_time, ROI) is the primary key of the input vector;
- A player can generate multiple vectors to rating system even for same image;
- The **event_time** is the capture time of the satellite image.

Rating Model

For *a certain image* *img* *at time* *t*,
Rating: player i -> player j:

$$w_{ij} = \sum_{\mathrm{ROI} \in \mathrm{ROIs}} \frac{\mathrm{ROI}(i,j)}{\mathrm{ROI}(i)} \times \frac{Cov(\mathrm{tags}_{\mathrm{ROI}}(i), \mathrm{tags}_{\mathrm{ROI}}(j))}{\mathrm{var}(\mathrm{tags}_{\mathrm{ROI}}(i))\mathrm{var}(\mathrm{tags}_{\mathrm{ROI}}(j))} \geq 0$$



Normalized Adjacency Matrix:

$$A = (\frac{w_{ij}}{\sum_j w_{ij}})$$

Obviously, *A* is **irreducible, real, non-negative, column-stochastic, and diagonal element being positive,** then eigenvalue of *A* is the player ***trust value***.

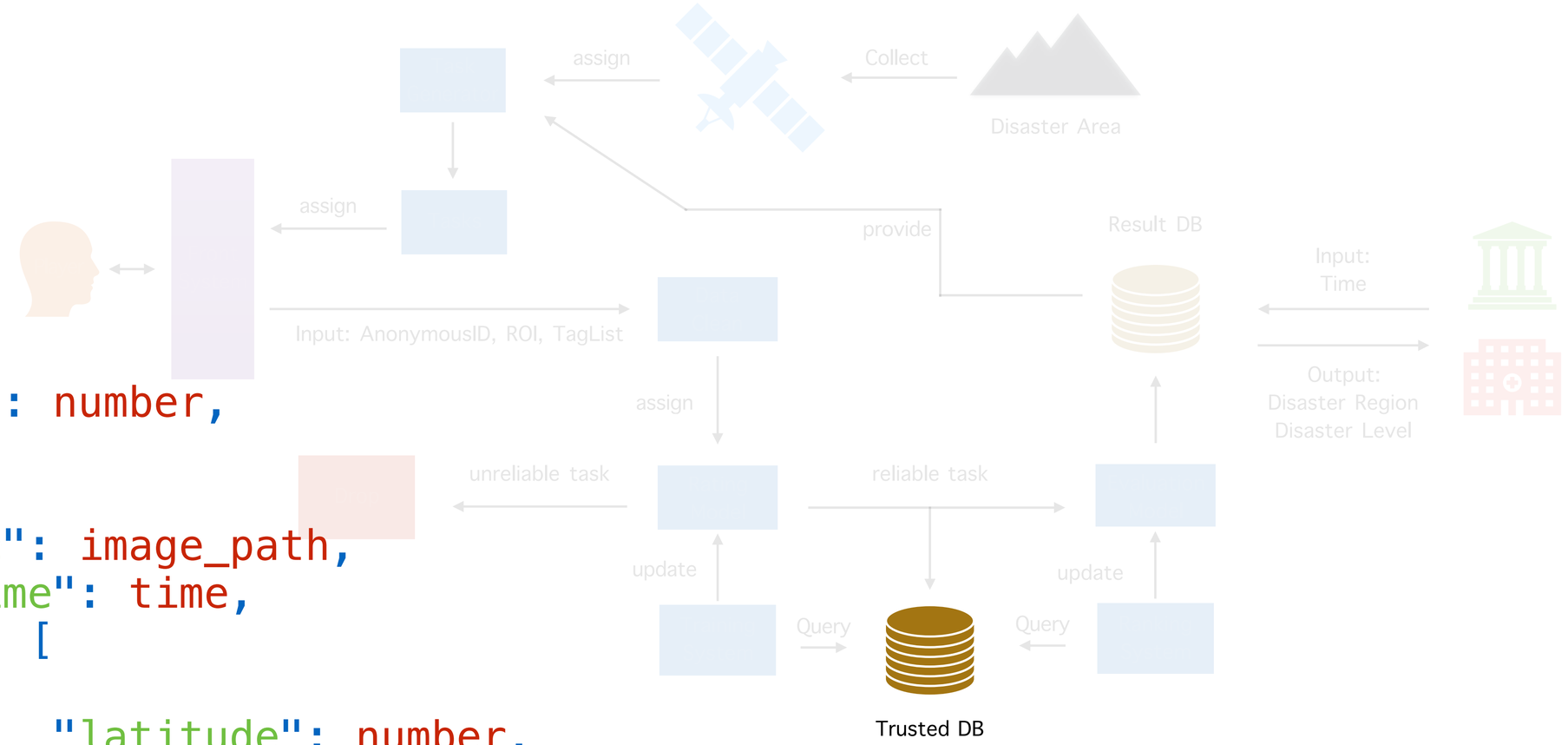
When a new player tagging task need to be rated,

- which means we need introduce a new node to the graph
- need calculate the trust value of new graph
- let *t'* is the trust value of new player
- if *t' >= mean(old_eigenvalues)*, then it is a reliable player, otherwise drop it.

Database Design

Trusted DB **Fields:**

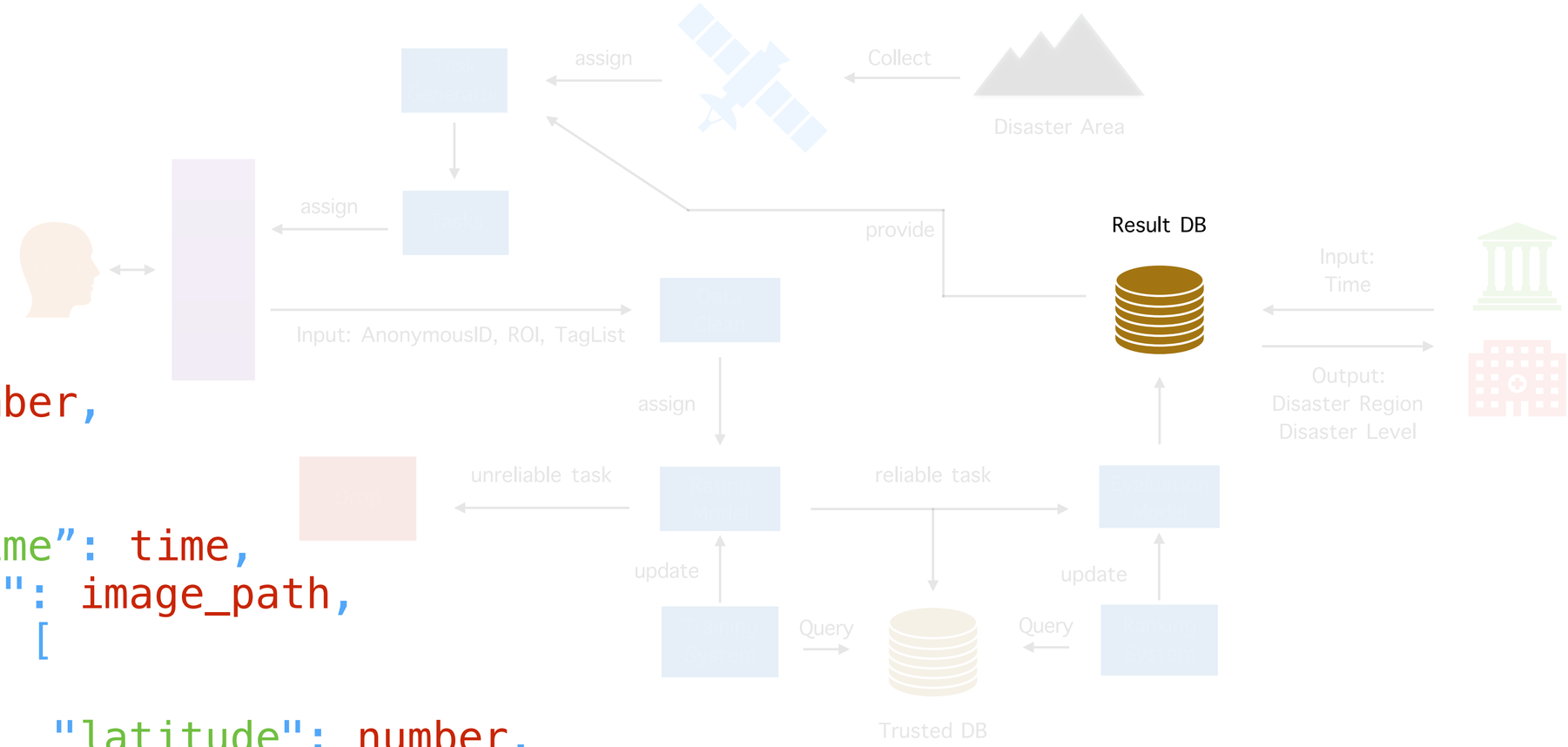
```
[
    {
        "anonymous_id": number,
        "tasks": [
            {
                "image": image_path,
                "at_time": time,
                "ROI": [
                    {
                        "latitude": number,
                        "longitude": number,
                        "tags": [tag1, tag2, ...]
                    }
                ]
            }
        ]
        "trust_value": number
    }
]
```



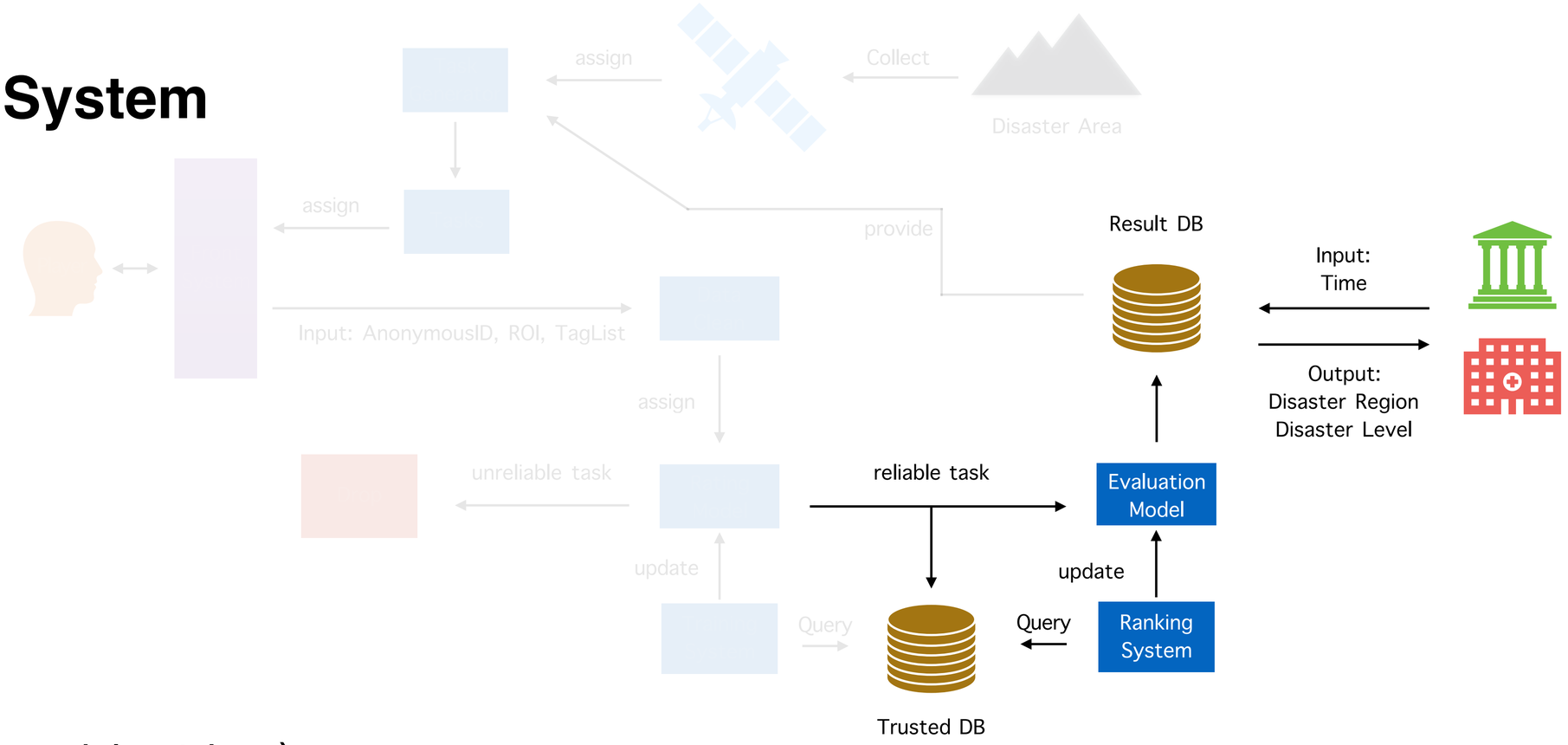
Database Design

Result DB **Fields:**

```
[
    {
        “area_id": number,
        "history": [
            {
                “at_time”: time,
                "image": image_path,
                "ROI": [
                    {
                        "latitude": number,
                        “longitude": number,
                        "tags": [tag1, tag2, ...]
                    }
                ],
                "disaster_level": number
            }
        ]
    }
]
```



Disaster Evaluation System



Query input:
- (time)/(area_id)/(area_id, time)

Model output:
- (area_id, time, disaster_level)

Note that:
- All results are evaluated from reliable tasks
- Evaluation Model generated by all reliable history

Disaster Evaluation Model

Now we have trusted results, each area has its tagging history.

For an area at time t, define ***disaster level*** as follows:

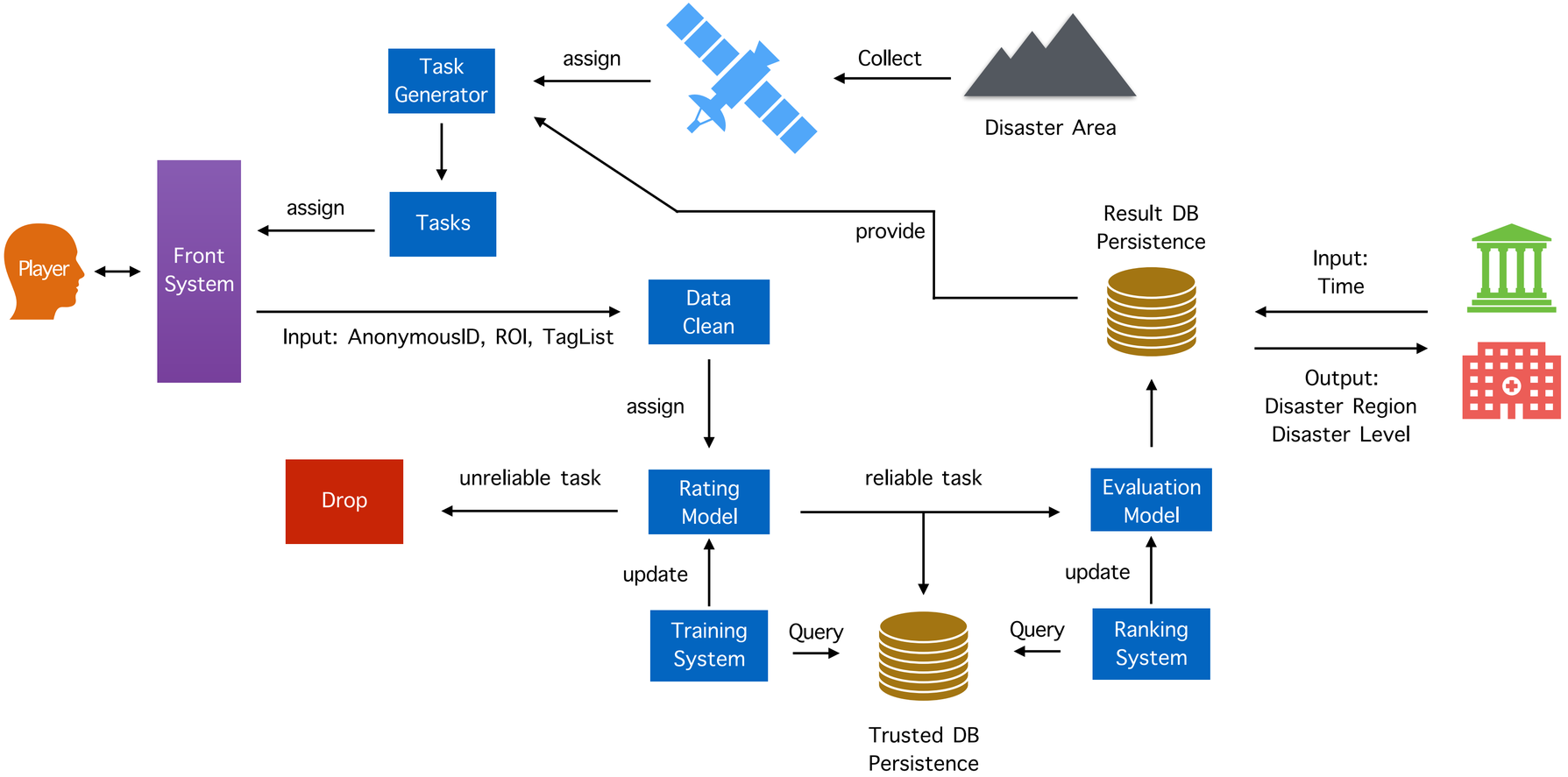
$$v_{area} = \frac{\sum_{\text{tag} \in \text{tags}} w_{tag} \times \#(\text{tag})}{\sum_{area \in areas} \sum_{\text{tag} \in \text{tags}} w_{tag} \times \#(\text{tag})}$$

where *w_{tag}* is pre-defined weight by system, *#(tag)* is the occur number of a tag.

Return value:

- disaster region: $\cup_{ROI \in ROIs} ROI$
- disaster level: v_{area}

TL; DR



1. Task Generator combines trusted results assign to players;
2. Always treat player as new player, but integrated as old player if exists;
3. Use ROI matching rate as graph edge weight, eigenvalue as trust value of player;
4. Disaster Evaluation use pre-defined weight, then defined the disaster level

FAQ